

What is claimed is:

1 1: A method of managing a lock utilized by a thread comprising:
2 selecting an action to perform upon the lock, wherein the action is selected from a
3 group comprising:
4 acquiring the lock,
5 trying to acquire the lock, and
6 releasing the lock;
7 asynchronously querying the current state of a lock, having a multi-value state;
8 speculatively determining the next state of the lock; and
9 attempting to transition the lock from the queried current state to the speculatively
10 determined next state.

1 2: The method of claim 1, further including, if the state transition fails and if the selected
2 action was either acquiring or releasing the lock, repeating, until the state transition
3 succeeds:
4 asynchronously querying the current state of the lock;
5 speculatively determining the next state of the lock;
6 attempting to transition the lock from the queried current state to the speculatively
7 determined next state.

1 3: The method of claim 2, further including, if
2 the state transition succeeds,

3 the selected action is acquiring the lock, and
4 the speculatively determined next state represents the acquisition of the lock,
5 indicating the acquisition of the lock.

1 4: The method of claim 3, further including, if
2 the state transition succeeds,
3 the selected action is acquiring the lock, and
4 the speculatively determined next state does not represent the acquisition of the
5 lock,
6 adding the thread to the end of a queue of threads waiting to acquire the lock;
7 waiting to receive notification that the thread may acquire the lock; and
8 indicating the acquisition of the lock.

1 5: The method of claim 2, further including, if
2 the state transition succeeds, and
3 the selected action is releasing the lock,
4 determining the number of threads in a queue to acquire the lock utilizing the
5 speculatively determined next state of the lock.

1 6: The method of claim 5, further including, if the queue includes at least a first thread,
2 removing the first thread from the queue; and
3 notifying the first thread that the first thread has acquired the lock.

1 7: The method of claim 1, further including, if
2 the selected action is trying to acquire the lock and
3 the state transition fails,
4 indicating that the lock was unable to be acquired.

1 8: The method of claim 1, further including, if
2 the state transition succeeds and
3 the selected action is trying to acquire the lock,
4 indicating the acquisition of the lock.

1 9: The method of claim 1, further including, if
2 the state transition succeeds,
3 the selected action is acquiring the lock, and
4 the speculatively determined next state represents the acquisition of the lock,
5 indicating the acquisition of the lock.

1 10: The method of claim 9, further including, if
2 the state transition succeeds,
3 the selected action is acquiring the lock, and
4 the speculatively determined next state does not represent the acquisition of the
5 lock,
6 adding the thread to the end of a queue of threads waiting to acquire the lock;
7 waiting to receive notification that the thread may acquire the lock; and
8 indicating the acquisition of the lock.

1 11: The method of claim 1, further including, if
2 the state transition succeeds, and
3 the selected action is releasing the lock,
4 determining the number of threads in a queue to acquire the lock utilizing the
5 speculatively determined next state of the lock.

1 12: The method of claim 11, further including, if the queue includes at least a first
2 thread,
3 removing the first thread from the queue; and
4 notifying the first thread that the first thread has acquired the lock.

1 13: The method of claim 1, wherein the thread includes:

2 a unique thread identifier;

3 a next thread field to facilitate access to the next thread in a queue of threads

4 waiting to acquire the lock; and

5 the thread is only capable of waiting for a single lock at a time.

1 14: The method of claim 1, wherein the action of acquiring the lock includes the inability

2 to timeout or fail to acquire the lock.

1 15: The method of claim 1, wherein the lock's current state may change between

2 asynchronously querying the current state of the lock; and

3 attempting to transition the lock from the queried current state to the speculatively

4 determined next state.

1 16: An apparatus comprising:

2 a lock, having a multi-state value, including:

3 a flag value, a first thread value, and a last thread value; and

4 a lock acquirer, which is capable of performing an acquisition of the lock via

5 asynchronously querying the current state of the lock;

6 speculatively determining the next state of the lock; and

7 attempting to transition the lock from the queried current state to the
8 speculatively determined next state.

1 17: The apparatus of claim 16, wherein the lock acquirer is further capable of performing
2 two general actions, including acquiring the lock, trying to acquire the lock; and
3 wherein, if the state transition fails and the general action is acquiring the lock,
4 the lock acquirer is further capable of, repeating, until the state transaction succeeds:
5 asynchronously querying the current state of the lock;
6 speculatively determining the next state of the lock; and
7 attempting to transition the lock from the queried current state to the speculatively
8 determined next state.

1 18: The apparatus of claim 17, wherein, if the state transition fails and the general action
2 is trying to acquire the lock, the lock acquirer is further capable of,
3 indicating that the lock was unable to be acquired.

1 19: The apparatus of claim 18, wherein, if the state transition succeeds and the general
2 action is trying to acquire the lock, the lock acquirer is further capable of,
3 indicating that the lock was acquired.

1 20: The apparatus of claim 16, wherein, if
2 the state transition succeeds,
3 the general action is acquire the lock, and
4 the speculatively determined next state represents the acquisition of the lock,
5 the lock acquirer is further capable of,
6 indicating that the lock was acquired.

1 21: The apparatus of claim 20, wherein, if
2 the state transition fails,
3 the general action is acquire the lock, and
4 the speculatively determined next state does not represent the acquisition of the
5 lock,
6 the lock acquirer is further capable of,
7 adding the thread to the end of a queue of threads waiting to acquire the lock;
8 waiting to receive notification that the thread may acquire the lock; and
9 indicating the acquisition of the lock.

1 22: The apparatus of claim 21, wherein the lock acquirer is unable to timeout or fail if
2 the selected general action is acquiring the lock.

1 23: An apparatus comprising:
2 a lock, having a multi-state value, including:
3 a flag value, a first thread value, and a last thread value; and
4 a lock releaser, which is capable of releasing a hold on the lock via
5 asynchronously querying the current state of the lock;
6 speculatively determining the next state of the lock; and
7 attempting to transition the lock from the queried current state to the
8 speculatively determined next state.

1 24: The apparatus of claim 23, wherein, if the state transition fails, the lock releaser is
2 further capable of, repeating, until the state transaction succeeds:
3 asynchronously querying the current state of the lock;
4 speculatively determining the next state of the lock; and
5 attempting to transition the lock from the queried current state to the speculatively
6 determined next state.

1 25: The apparatus of claim 23, wherein, if the state transition succeeds, the lock releaser
2 is further capable of determining the number of threads in a queue of threads waiting to
3 acquire the lock utilizing the speculatively determined next state of the lock.

1 26: The apparatus of claim 25, wherein, if the queue includes at least a first thread, the
2 lock releaser is further capable of:

3 removing the first thread from the queue; and

4 notifying the first thread that the first thread has acquired the lock.

1 27: The apparatus of claim 26, wherein the lock releaser is capable of removing the first
2 thread from the queue utilizing a thread having:

3 a unique thread identifier; and

4 a next thread value to facilitate access to the next thread in the queue.

1 28: The apparatus of claim 23, wherein the lock is capable of changing state in between
2 the time the lock releaser

3 asynchronously queries the current state of the lock; and

4 attempts to transition the lock from the queried current state to the speculatively
5 determined next state.

1 29: An article comprising:

2 a storage medium having a plurality of machine accessible instructions, wherein when the
3 instructions are executed, the instructions provide for:

4 selecting an action to perform upon a lock utilized by a thread, wherein the action

5 is selected from a group comprising:

6 acquiring the lock,
7 trying to acquire the lock, and
8 releasing the lock;
9 asynchronously querying the current state of a lock, having a multi-value state;
10 speculatively determining the next state of the lock; and
11 attempting to transition the lock from the queried current state to the speculatively
12 determined next state.

1 30: The article of claim 29, further including instructions providing for, if the state
2 transition fails and if the selected action was either acquiring or releasing the lock,
3 repeating, until the state transition succeeds:
4 asynchronously querying the current state of the lock;
5 speculatively determining the next state of the lock;
6 attempting to transition the lock from the queried current state to the speculatively
7 determined next state.

1 31: The article of claim 30, further including instructions providing for, if
2 the state transition succeeds,
3 the selected action is acquiring the lock, and
4 the speculatively determined next state represents the acquisition of the lock,
5 indicating the acquisition of the lock.

1 32: The article of claim 31, further including instructions providing for, if
2 the state transition succeeds,
3 the selected action is acquiring the lock, and
4 the speculatively determined next state does not represent the acquisition of the
5 lock,
6 adding the thread to the end of a queue of threads waiting to acquire the lock;
7 waiting to receive notification that the thread may acquire the lock; and
8 indicating the acquisition of the lock.

1 33: The article of claim 30, further including instructions providing for, if
2 the state transition succeeds, and
3 the selected action is releasing the lock,
4 determining the number of threads in a queue to acquire the lock utilizing the
5 speculatively determined next state of the lock.

1 34: The article of claim 33, further including instructions providing for, if the queue
2 includes at least a first thread,
3 removing the first thread from the queue; and
4 notifying the first thread that the first thread has acquired the lock.

1 35: The article of claim 29, further including instructions providing for, if
2 the selected action is trying to acquire the lock and
3 the state transition fails,
4 indicating that the lock was unable to be acquired.

1 36: The article of claim 29, further including instructions providing for, if
2 the state transition succeeds and
3 the selected action is trying to acquire the lock,
4 indicating the acquisition of the lock.

1 37: The article of claim 29, further including instructions providing for, if
2 the state transition succeeds,
3 the selected action is acquiring the lock, and
4 the speculatively determined next state represents the acquisition of the lock,
5 indicating the acquisition of the lock.

1 38: The article of claim 37, further including instructions providing for, if
2 the state transition succeeds,
3 the selected action is acquiring the lock, and
4 the speculatively determined next state does not represent the acquisition of the
5 lock,

6 adding the thread to the end of a queue of threads waiting to acquire the lock;
7 waiting to receive notification that the thread may acquire the lock; and
8 indicating the acquisition of the lock.

1 39: The article of claim 29, further including instructions providing for, if
2 the state transition succeeds, and
3 the selected action is releasing the lock,
4 determining the number of threads in a queue to acquire the lock utilizing the
5 speculatively determined next state of the lock.

1 40: The article of claim 39, further including instructions providing for, if the queue
2 includes at least a first thread,
3 removing the first thread from the queue; and
4 notifying the first thread that the first thread has acquired the lock.

1 41: The article of claim 29, wherein the thread includes:
2 a unique thread identifier;
3 a next thread field to facilitate access to the next thread in a queue of threads
4 waiting to acquire the lock; and
5 the thread is only capable of waiting for a single lock at a time.

1 42: The article of claim 29, wherein the action of acquiring the lock includes the inability
2 to timeout or fail to acquire the lock.

1 43: The article of claim 29, wherein the lock's current state may change between
2 asynchronously querying the current state of the lock; and
3 attempting to transition the lock from the queried current state to the speculatively
4 determined next state.

1 44: A system comprising:
2 a memory element, capable of storing a queue of threads, each thread including
3 a unique thread identifier, and a next thread value to facilitate access to the
4 next thread in the queue;
5 a lock, having a multi-state value, including:
6 a flag value, a first thread value, and a last thread value; and
7 a lock acquirer, which is capable of performing an acquisition of the lock via
8 asynchronously querying the current state of the lock;
9 speculatively determining the next state of the lock; and
10 attempting to transition the lock from the queried current state to the
11 speculatively determined next state.

1 45: The system of claim 44, wherein the lock acquirer is further capable of performing
2 two general actions, including acquiring the lock, trying to acquire the lock; and
3 wherein, if the state transition fails and the general action is acquiring the lock,
4 the lock acquirer is further capable of, repeating, until the state transaction succeeds:
5 asynchronously querying the current state of the lock;
6 speculatively determining the next state of the lock; and
7 attempting to transition the lock from the queried current state to the speculatively
8 determined next state.

1 46: The system of claim 45, wherein, if the state transition fails and the general action is
2 trying to acquire the lock, the lock acquirer is further capable of,
3 indicating that the lock was unable to be acquired.

1 47: The system of claim 46, wherein, if the state transition succeeds and the general
2 action is trying to acquire the lock, the lock acquirer is further capable of,
3 indicating that the lock was acquired.

1 48: The system of claim 44, wherein, if
2 the state transition succeeds,
3 the general action is acquire the lock, and
4 the speculatively determined next state represents the acquisition of the lock,

5 the lock acquirer is further capable of,
6 indicating that the lock was acquired.

1 49: The system of claim 48, wherein, if
2 the state transition fails,
3 the general action is acquire the lock, and
4 the speculatively determined next state does not represent the acquisition of the
5 lock,
6 the lock acquirer is further capable of,
7 adding the thread to the end of the queue of threads waiting to acquire the lock;
8 waiting to receive notification that the thread may acquire the lock; and
9 indicating the acquisition of the lock.

1 50: The system of claim 49, wherein the lock acquirer is unable to timeout or fail if the
2 selected general action is acquiring the lock.

1 51: A system comprising:
2 a memory element, capable of storing a queue of threads, each thread including
3 a unique thread identifier, and a next thread value to facilitate access to the
4 next thread in the queue;
5 a lock, having a multi-state value, including:

6 a flag value, a first thread value, and a last thread value; and
7 a lock releaser, which is capable of releasing a hold on the lock via
8 asynchronously querying the current state of the lock;
9 speculatively determining the next state of the lock; and
10 attempting to transition the lock from the queried current state to the
11 speculatively determined next state.

1 52: The system of claim 51, wherein, if the state transition fails, the lock releaser is
2 further capable of, repeating, until the state transaction succeeds:
3 asynchronously querying the current state of the lock;
4 speculatively determining the next state of the lock; and
5 attempting to transition the lock from the queried current state to the speculatively
6 determined next state.

1 53: The system of claim 51, wherein, if the state transition succeeds, the lock releaser is
2 further capable of determining the number of threads in the queue of threads waiting to
3 acquire the lock utilizing the speculatively determined next state of the lock.

1 54: The system of claim 53, wherein, if the queue includes at least a first thread, the lock
2 releaser is further capable of:

3 removing the first thread from the queue; and
4 notifying the first thread that the first thread has acquired the lock.

1 55: The system of claim 51, wherein the lock is capable of changing state in between the
2 time the lock releaser
3 asynchronously queries the current state of the lock; and
4 attempts to transition the lock from the queried current state to the speculatively
5 determined next state.